REMARKS

Claims 1–20 are pending in the present application.

Reconsideration of the claims is respectfully requested.

35 U.S.C. § 102 (Anticipation)

Claims 1–13 were rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent No. 4,991,080 to *Emma et al.* This rejection is respectfully traversed.

A claim is anticipated only if each and every element is found, either expressly or inherently described, in a single prior art reference. The identical invention must be shown in as complete detail as is contained in the claim. MPEP § 2131 at p. 2100-76 (8th ed. rev. 3 August 2005).

Independent claims 1, 8 and 14 each recite (a) branching and non-branching clusters both capable of executing instructions and computing branch conditions, but with the branching cluster performing branch address computations for both the branching and non-branching cluster because the non-branching cluster is incapable of performing branch address computations, and (b) remote conditional branching control circuitry that causes the branching cluster to perform a branch address computation in response to sensing a conditional branch instruction in the non-branching cluster. Such a combination of features is not found in the cited references.

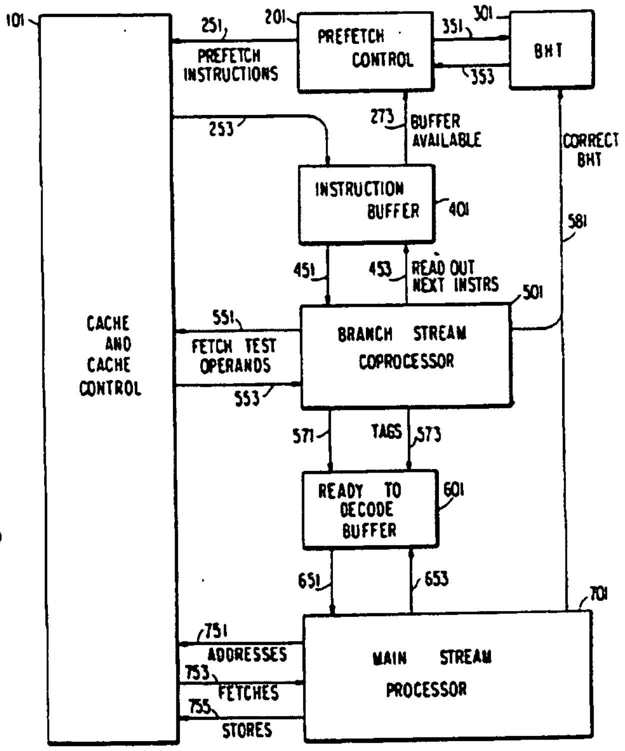
The Office Action states:

> In Emma, the branching cluster comprises elements 301 and 501. The non-branching cluster comprises elements 201, 401, 601, and 701. The branch history table, element 301, is a part of the branching cluster. The branch history table performs branch address computations, such as branch address history corrections1updates (column

> 5, line 57-column 6, line 22, column 7, lines 22-36, column 22, lines 55-58, column 23, line 63-column 24, line 16). The non-branching cluster does not include a branch history table, so the non-branching cluster is necessarily incapable of performing branch address computations, such as the branch address history corrections l updates that the branch history table computes in the branching cluster. So the non-branching cluster of Emma is in fact incapable of performing branch address computations.

Paper No. 101905, page 3. This analysis arbitrarily selects elements from *Emma* to fit the template of the claims, without reference to the actual teachings of *Emma et al. Emma et al* does not logically reserve the branch history table (BHT) 301 to exclusive association with the branch stream coprocessor 501 as asserted in the Office Action. Figure 3 of *Emma et al* depicts a processing system including a branch stream coprocessor 501 and a main stream processor 701:

FIG.3



Emma et al, Figure 3. Branch stream coprocessor 501 (a branch prediction unit) does NOT execute instructions, but instead merely partially "pre-executes" selected instructions to determine branch conditions from operands, before forwarding the partially executed instructions to the main stream processor 701 with status information for complete execution:

> In order to pre-execute instructions received two at a time from the instruction buffer 401, the branch stream coprocessor 501 fetches corresponding test operands, when appropriate, from the cache 101 over a line 551. The operands are sent from the cache over a line 553. It should be noted that the branch stream coprocessor 501 will not pre-execute all instructions, but only those which require some action before being executed by a main stream processor 701. Other instructions pass directly to a ready-to-decode buffer 601. Because instructions which are pre-executed are delayed before passage to the ready-to-decode buffer 601, a count is kept in the coprocessor, so that pre-executed instructions occupy the appropriate place in the ready-to-decode buffer 601.
>
> All instructions pass from the branch stream coprocessor 501 over a line 571 to the ready-to-decode buffer 601. Those instructions which have been pre-executed will be accompanied by additional bits, output over a line 573, which indicate to the main stream processor 701 what has been done, and what remains to be done. If the branch stream coprocessor 501 detects an error in the branch history table 301, a correction is output over a line 581 to the branch history table 301.

Emma et al, column 9, line 65 through column 10, line 19. Thus the branch stream coprocessor 501 in *Emma et al* is not capable of executing instructions as required of the branching cluster in the claims, but is instead merely capable of computing branch conditions, only.

In addition, the branch history table 301 is not exclusively associated with the branch stream coprocessor 501 as asserted in the Office Action. The main stream processor 701 in *Emma et al* employs branch history table 301:

> If the main stream processor 701 is ***required to resolve a branch*** which the branch stream coprocessor 501 was unable to handle and correction of the branch history table 301 is required as a result, correction is output over a line 581 to the branch history table 301.

Emma et al, column 10, lines 34–38 (emphasis added). Thus, contrary to the assertion within the Office Action, main stream processor 701 is capable of performing branch address computations, not incapable of performing such computations as required by the claims.

Still further, prefetch control 201 and instruction buffer 401 operate to fetch and buffer instructions based on information within branch history table 301, providing instructions to branch stream coprocessor 501. *Emma et al*, column 9, lines 27–40. Thus, prefetch control 201 and instruction buffer 401 are not uniquely associated with main stream processor 701 as asserted in the Office Action, but instead operates in conjunction with branch stream coprocessor 501 as well. Branch history table 301 also operates in conjunction with both prefetch control 201 and instruction buffer 401, and is not operable only with branch stream coprocessor 501 to the exclusion of prefetch control 201, instruction buffer 401, ready-to-decode buffer 601 and main stream processor 701 as asserted in the Office Action.

Finally, independent claims 1, 8 and 14 also recite remote conditional branching control circuitry that causes the branching cluster to perform a branch address computation in response to sensing a conditional branch instruction in the non-branching cluster. Such a feature is not found in the cited reference. The Office Action asserts that this limitation is satisfied by signal line 581 in Figure 3 of *Emma et al.* Paper No. 051205, page 3. In support of that assertion, the Office Action cites the following portions of *Emma et al* (cited portions quoted in context):

> Since a branch is appearing as the first instruction, the branch stream coprocessor 501 has not resolved it. Consequently, it will be necessary to check its actual target address from the address generation step 729 against the target address which the branch history table 301 has predicted. Also, it will be necessary to compare the actual action taken, as determined in the execute and putaway step 735, against the action which the branch history table 301 predicted. In other respects, the branch proceeds conventionally through the pipeline. The pipeline steps are as follows:
> a. Decode step 717 -- A signal that this instruction is a branch is sent via a path 757 to a correction handling stack 743. The instruction address also is sent, on a path 759. The correction handling stack 743 locates the predicted target address for this branch, and, when the actual target address is available from the address generation step 729 on a path 761, a comparison is carried out between the two. If they match, the prediction was correct, and no branch history table correction is needed. However, if there is no match, the branch history table 301 contains an error. This error is signalled on a path 581 to the branch history table 301 (see FIG. 17). In this case, all instructions following the wrongly predicted branch, in the pipeline, the ready-to-decode buffer 601, and the instruction buffer 401, are invalid. The branch history table 301 must be corrected, and instruction prefetching must resume with the correct target address. This procedure will be described in greater detail below.
>
> . . . .
>
> Branch history table correction now will be described, with reference to FIG. 17. Such correction becomes necessary either through pre-execution activity of the branch stream coprocessor 501, or through subsequent regular execution activity of the main stream processor 701 with regard to branches which the branch stream coprocessor 501 was unable to resolve.

Emma et al, column 22, lines 29–58, column 23, line 24–30. However, as apparent from the above-quoted text, signal line 581 merely conveys a signal and is not capable of "sensing a conditional branch instruction in the non-branching cluster" as required by the claims. It should be noted that the cited portion of *Emma et al* relates to the operation of the pipeline in main stream processor 701 when the first instruction is a branch instruction. In the system of *Emma et al*, main stream processor 701 "senses" a conditional branch instruction within itself, having received a conditional branch instruction from branch stream coprocessor 501 that was "not resolved" by branch stream coprocessor 501. Moreover, signal line 581 merely signals an error or nonmatching condition between the predicted target address in the branch history table 301 and the actual target address computed during execution of the conditional branch instruction within main stream processor 701. Signal line 581, when asserted, prompts an update of branch history table 301, but does not trigger performance of a branch address computation within the branch stream coprocessor 501 and/or branch history table 301 (the asserted "branching cluster") as recited in the claims.

Therefore, the rejection of claims 1–13 under 35 U.S.C. § 102 has been overcome.

35 U.S.C. § 103 (Obviousness)

Claims 14–20 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Emma et al* in view of U.S. Patent No. 4,777,589 to *Boettner et al.* This rejection is respectfully traversed.

In *ex parte* examination of patent applications, the Patent Office bears the burden of establishing a *prima facie* case of obviousness. MPEP § 2142, p. 2100-133 (8th ed. rev. 3 August 2005). Absent such a *prima facie* case, the applicant is under no obligation to produce evidence of nonobviousness. *Id.*

To establish a *prima facie* case of obviousness, three basic criteria must be met: First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure. *Id.*

As noted above, independent claim 14 recites features not found in *Emma et al.* Such features are also not found in *Boettner et al.*

Therefore, the rejection of claims 14–20 under 35 U.S.C. § 103 has been overcome.

If any issues arise, or if the Examiner has any suggestions for expediting allowance of this Application, the Applicant respectfully invites the Examiner to contact the undersigned at the telephone number indicated below or at ***dvenglarik@davismunck.com***.

The Commissioner is hereby authorized to charge any additional fees connected with this communication or credit any overpayment to Deposit Account No. 50-0208.

Respectfully submitted,

DAVIS MUNCK, P.C.

[illegible]

Date: 3-1-06

Daniel E. Venglarik
Registration No. 39,409

P.O. Box 802432
Dallas, Texas 75380
(972) 628-3621 (direct dial)
(972) 628-3600 (main number)
(972) 628-3616 (fax)
E-mail: *dvenglarik@davismunck.com*